AFTO

PTO/SB/21 (09-06)
Approved for use through 03/31/2007. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Total Number of Pages in This Submission: 73

Application Number	10/614,347
Filing Date	July 8, 2003
First Named Inventor	Gregory A. Becker et al.
Art Unit	2166
Examiner Name	Khanh B. Pham
Attorney Docket Number	PA3166US

ENCLOSURES *(Check all that apply)*

- ☐ Fee Transmittal Form
  - ☐ Fee Attached
- ☐ Amendment/Reply
  - ☐ After Final
  - ☐ Affidavits/declaration(s)
- ☐ Extension of Time Request
- ☐ Express Abandonment Request
- ☐ Information Disclosure Statement
- ☐ Certified Copy of Priority Document(s)
- ☐ Reply to Missing Parts/ Incomplete Application
  - ☐ Reply to Missing Parts under 37 CFR 1.52 or 1.53
- ☐ Drawing(s)
- ☐ Licensing-related Papers
- ☐ Petition
- ☐ Petition to Convert to a Provisional Application
- ☐ Power of Attorney, Revocation Change of Correspondence Address
- ☐ Terminal Disclaimer
- ☐ Request for Refund
- ☐ CD, Number of CD(s) _____
  - ☐ Landscape Table on CD
- ☐ After Allowance Communication to TC
- ☐ Appeal Communication to Board of Appeals and Interferences
- ☐ Appeal Communication to TC **(Appeal Notice, Brief, Reply Brief)**
- ☐ Proprietary Information
- ☐ Status Letter
- ☑ Other Enclosure(s) (please Identify below):
  1) Confirmation Postcard;
  2) Response to Non-Compliant Appeal Brief ("Brief on Appeal," 24 pages, in triplicate)

Remarks

Total page number does not include postcard.

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm Name	Carr & Ferrell LLP Customer Number 22830		
Signature	Kenneth M. Kaslow		
Printed name	Kenneth M. Kaslow		
Date	February 15, 2007	Reg. No.	32,246

CERTIFICATE OF TRANSMISSION/MAILING

I hereby certify that this correspondence is being facsimile transmitted to the USPTO or deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on the date shown below:

Signature	Kenneth M. Kaslow		
Typed or printed name	Kenneth M. Kaslow, Reg. No. 32,246	Date	February 15, 2007

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

APPLICANTS:	Gregory A. Becker et al.
SERIAL NO.:	10/614,347
FILING DATE:	July 8, 2003
TITLE:	System and Method for Backing Up a Computer System
EXAMINER:	Khanh B. Pham
ART UNIT:	2166
ATTY. DKT. NO:	PA3166US

CERTIFICATE OF MAILING

I hereby certify that this paper is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Mail Stop Appeal Brief, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on the date printed below:

Date: 2/15/07 Kenneth M. Kaslow

Kenneth M. Kaslow

Mail Stop Appeal Brief -- Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

BRIEF ON APPEAL

Sir:

This brief is submitted in an Appeal from the Final Office action of June 30, 2006 and the Notice of Panel Decision from Pre-Appeal Brief Review of September 22, 2006, rejecting claims 1-12, 14-15, 20, and 26-32 of the above-referenced patent application, and in response to the Notification of Non-Compliant Appeal Brief mailed January 17, 2007.

The fees required under § 1.17, and any required petition for extension of time for filing this brief and fees therefor, are dealt with in the accompanying TRANSMITTAL OF APPEAL BRIEF.

This brief contains these items under the following headings, and in the order set forth below (37 C.F.R. § 41.37(c)(i)):

I REAL PARTY IN INTEREST
II RELATED APPEALS AND INTERFERENCES
III STATUS OF CLAIMS
IV STATUS OF AMENDMENTS
V SUMMARY OF CLAIMED SUBJECT MATTER
VI GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL
VII ARGUMENT
VIII CLAIMS APPENDIX
IX EVIDENCE APPENDIX
X RELATED PROCEEDING APPENDIX

The final page of this brief bears the practitioner's signature.

(I) Real Party in Interest (37 C.F.R. § 41.37(c)(1)(i))

The real party in interest in the above-referenced patent application is Mendocino Software, Inc. of Fremont, California.

(II) Related Appeals and Interferences (37 C.F.R. § 41.37(c)(1)(ii))

With respect to other prior or pending appeals, interferences, or related judicial proceedings that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, there are no other such appeals, interferences, or related judicial proceedings.

(III) Status of Claims (37 C.F.R. § 41.37(c)(1)(iii))

(a) Total Number of Claims in Application

22 claims are pending.

(b) Status of All the Claims in Application

i. Claims withdrawn from consideration: Claims 16-19 and 21-25

ii. Claims pending: Claims 1-12, 14-15, 20, and 26-32

iii. Claims rejected: Claims 1-12, 14-15, 20, and 26-32

(c) Claims on Appeal

Claims 1-12, 14-15, 20, and 26-32 are being appealed.

(IV) Status of Amendments (37 C.F.R. § 41.37(c)(1)(iv))

An amendment was made to 4, 8, 14, and 15 in response to objections by the Examiner. Claims 16-19 and 21-25 were cancelled and claims 1-12, 14-15, and 20 were elected for prosecution in response to in response to the first office action issued on

February 16, 2006. No substantive changes were made to the claims via the amendment. Claims 26-32 were added. On June 30, 2006, an Advisory Action was sent indicating that the rejection of elected claims 1-12, 14-15, 20, and new claims 26-32 were rejected under 35 U.S.C. §102(e) as being anticipated by *Goldstein* (US 6,665,815).

As to the status of any amendment filed subsequent to final rejection, there are no such amendments after final.

(V) Summary of Claimed Subject Matter (37 C.F.R. § 41.37(c)(1)(v))

With respect to a summary of claim 1, a method for maintaining a backup storage system for a data storage system is provided (See FIG. 3). A plurality of data writes from are received from an application program, the plurality of data writes occurring between a first time and a second time. A backward increment between data on the data storage system at the second time and data on the data storage system at the first time is determined based on the plurality of data writes from the application program to the data storage system. The backward increment is stored. The plurality of data writes is also stored. The backup storage system is then updated so that the data on the data storage system at the second time is the same as the data on the backup storage system at the second time (See, for example, paragraphs [0032] through [0037]).

With respect to a summary of claim 20, a method for using a backup storage system for a data storage system is provided (See FIG. 3). A plurality of data writes captured between an application and the data storage system are received, the plurality of data writes occurring between a first time and a second time. Data blocks in the data storage system that were changed are identified based on the plurality of data writes (See, for example, paragraph [0061] and [0070]). The plurality of data writes are applied to an image on the backup storage system (See, for example, paragraphs [0070], [0087],

and [0119]). A forward increment between data on the data storage system at the first time and data on the data storage system at the second time is determined based on the plurality of data writes. A backward increment between data on the data storage system at the second time and data on the data storage system at the first time is also determined based on a plurality of data writes. The forward increment and the backward increment are stored. The plurality of data writes is also stored. The backup storage system is then updated so that the data on the data storage system at the second time is the same as the data on the backup storage system at the second time (See, for example, paragraphs [0032] through [0037]).

With respect to a summary of claim 26, a system for maintaining a backup storage system for a data storage system is provided. A production intercept layer is configured to receive a plurality of data writes from an application program, the plurality of data writes occurring between a first time and a second time (See, for example, FIG. 2, 240, FIG. 3, 305, and paragraphs [0061] and [0062]). A backup agent is configured to determine a backward increment between data on the data storage system at the second time and data on the data storage system at the first time based on the plurality of data writes from the application program to the data storage system (See, for example, FIG. 3, 354, and paragraphs [0063] through [0067]). A log file container is configured to store the backward increment (See, for example, FIG. 3, 318, and paragraphs [0063] through [0067]). A storage device is configured to store the plurality of data writes (See, for example, paragraphs [0071] and [0072]). A backup manager is configured to update the backup storage system so that the data on the data storage system at the second time is the same as the data on the backup storage system at the second time (See, for example, FIG. 3, 350, and paragraphs [0074] through [0076]).

(VI) Grounds of Rejection to be Reviewed on Appeal (37 C.F.R. § 41.37(c)(1)(vi))

(a) The Examiner's Rejection

Claims 1-12, 14-15, 20, and 26-32 have been rejected under 35 U.S.C. §102(e) as being anticipated by *Goldstein* (US 6,665,815).

(VII) Argument (37 C.F.R. § 41.37(c)(1)(vii))

The following groups of claims are identified below.

(a) Group #1: Claims 1-12 and 14-15

(b) Group #2: Claims 20 and 26-32

Group #1 is different from group #2. Group #1 sets forth claims for maintaining a backup storage system for a data storage system while group #2 sets forth claims for using a backup storage system for a data storage system. Group #1 sets forth determining a backward increment and updating the backup storage system, while group #2 sets forth identifying changed data blocks as well as determining both a backward and a forward increment.

With regard to claim 1, the Examiner asserts that *Goldstein* teaches a method for maintaining a backup storage system for a data storage system comprising "receiving a plurality of data writes from an application program, the plurality of data writes occurring between a first time and a second time" at col. 5 lines 44-49 and Fig. 3; [and] "determining a backward increment between data on the data storage system the second time and data on the data storage system at the first time based on the plurality of data writes from the application program to the data storage system" at Col. 6 lines 6-60 and Fig. 7; "storing the backward increment" at Col. 6, lines 6-31; "storing the plurality of data writes" at Col. 6, lines 6-31; "and updating the backup storage system

so that the data on the data storage system at the second time is the same as the data on the backup storage system at the second time" at Col. 6, lines 6-31.

In the subsequent office action mailed June 30, 2006, the Examiner further asserts that "the difference between the two snapshots is the same as a plurality of data writes applied to the first snapshot because First Snapshot + Data Writes = Second Snapshot." The Examiner goes on to assert that "therefore... Second Snapshot – First Snapshot = Data Writes or... the difference between two snapshot = Plurality of Data Writes" and that "Goldstein therefore inherently anticipates the claimed limitation." (*Office action*, 7)

With respect to claim 20, the Examiner asserts that *Goldstein* teaches a method for using a backup storage system for a data storage system comprising "receiving a plurality of data writes captured between an application and the data storage system, the plurality of data writes occurring between a first time and a second time" at Col. 5 lines 44-48 and Fig. 3; "identifying data blocks in the data storage system that were changed based on the plurality of data writes" at Col 5 lines 23-48; "applying the plurality of data writes to an image on the backup storage system" at Col. 6 lines 6-31; "determining a forward increment between data on the data storage system at the first time and data on the data storage system at the second time based on the plurality of data writes" at Col. 3 line 55 to Col. 4 line 50 and Figs. 4,6; [and] "determining a backward increment between data on the data storage system at the second time and data on the data storage system at the first time based on a plurality of data writes" at Col. 6 lines 6-31 and Figs. 7-11; "storing the forward increment" at Col. 3 line 55 to col. 4 line 50; "storing the backward increment" at Col. 6 lines 6-31 and FIGS. 7-11; "storing the plurality of data writes" at Col. 6 lines 6-31 and FIGS. 7-11; "and updating the backup storage system so that the data on the data storage system at the second time is

the same as the data on the backup storage system at the second time." At Col. 6, lines 6-31 and FIGS. 7-11.

Goldstein et al. differs from the Present Application

In these rejections, the Examiner has failed to show that *Goldstein* anticipates claims 1-12, 14-15, 20, and 26-32. Applicants will show that the Examiner's assertions regarding *Goldstein* present clear errors of fact. Applicants hereby incorporate the previously submitted arguments filed in the Amendment mailed on May 16, 2006 and the Pre-Appeal Brief Request for Review filed on July 31, 2006.

The present application describes systems and methods for maintaining a backup storage system. An intercept agent continuously captures data writes from an application program. The data writes are utilized to recover data at any point in time. *Goldstein* cannot recover data to any point in time because *Goldstein* teaches incremental backups using snapshots. The snapshot is a virtual copy of a disk volume. A different list is also generated listing identifiers for data blocks in one snapshot that differ from data blocks in another snapshot. Because *Goldstein* takes snapshots of the disk volume and does not continuously received data writes, *Goldstein* cannot recover data at any point in time.

As an example, the invention taught by *Goldstein* equates to taking two pictures of a chair in a room at a first time and a second time. The difference list represents the distance between the chair locations at the first time and the second time. The chair, however, may have moved into another room, been moved hundreds of times, and so forth before the second picture was taken. Accordingly, the history of the chair movement is unknown. The claims of the present application may be represented by a video of the chair. The movement

of the chair from the first time to the second time is recorded and thus, the movement of the chair, the various locations of the chair, and so forth are known. Similarly, the data writes set forth in the claims of the present application are continuously captured. Accordingly, the data can be recovered at any point in time, unlike *Goldstein* , which can only recover data based on the time the snapshots are taken and the data blocks that have changed between the snapshots. Again, although the chair in the example may have only moved 10 feet in the two pictures (i.e. the data blocks between the two snapshots taken), the chair may have actually moved several hundred feet before the second picture was taken. Thus, although *Goldstein* takes various snapshots, the story of what happened to the data between each snapshot is, for the most part, unknown.

The Examiner appears to equate *Goldstein* to the present application because it is possible, under certain circumstances, for data recovered according to one of the snapshots taken in *Goldstein* to be similar to the data recovered at a particular point in time in the present application. An arbitrary time from *Goldstein* that happens to coincide to the same point in time according to the present application cannot simply be proposed to equate *Goldstein* with the present application. Just because a similar result may sometimes be accomplished does not support a finding that *Goldstein* anticipates the claims of the present invention. As discussed herein, the present application can recover data to any point in time based on continuous capture of the data writes, while *Goldstein* can only perform intermittent backups based on snapshots. Other than the difference between the two snapshots, *Goldstein* cannot determine what happened to the data between the two points in time represented by the

snapshots. Thus, *Goldstein* and the present application neither perform data recovery or backing up in the same matter nor do they achieve the same result.

Goldstein et al.

A careful review of *Goldstein* reveals that *Goldstein* takes two snapshots of a disk volume at two different times. *Goldstein* compares the two snapshots to determine a difference list, based on data blocks that differ between the two snapshots. In order to create a backup, *Goldstein* teaches that the data blocks from the difference list can be added to a succedent snapshot.

Specifically, *Goldstein* takes a snapshot of a disk volume at a base time and another snapshot of the disk volume at a first time. The two snapshots are virtual copies of the disk volume at those two different times. "[A] snapshot is a virtual copy of a disk volume." (col. 3 lines 43-44 and FIG. 2.) The snapshots capture the state of the volume. That is, the snapshots are made only when the system is in a "consistent state." (col. 3, lines 56-67 and FIG. 3) A snapshot of a disk is static capture of the entire contents of a volume at a single point in time.

Goldstein bases a 'snapshot difference list' on the difference between the state of the disk at the base time and the state of the disk at the first time. At Col. 4 lines 13-18 *Goldstein* describes obtaining a 'difference list'. "A first succedent snapshot difference list 121 (S_{01}) in data volume state snapshots is then obtained. A 'snapshot difference list' (e.g., $S_0 -> S_1$) is a list of identifiers of those data blocks in the first state snapshot 113 (S_1) that differ from the data blocks in the base state snapshot 111 (S_0)." (See also Col. 4 lines 28-36)

The Present Application

The present application, as discussed herein, sets for systems and methods for maintaining and using backup storage systems. Data writes are continuously captured (specification [0089]) in transit from an application to a set of storage devices (specification [0033]). The data writes may be copied, stored, and so forth, according to various embodiments.

The captured data writes are processed for later use (specification [0034]). The data writes may be stored according to a chronologically ordered sequence using meta information. A backward increment from a second point in time to a first point in time and/or a forward increment from a first point in time to a second point in time can be determined based on the data writes and the meta information (specification [0035] through [0036]). Accordingly, data can be reconstructed or recovered at any point in time (FIG. 3; see also specification [0056] through [0089]).

Claim 1

Claim 1, of the present application, recites "receiving a plurality of data writes from an application program, the plurality of data writes occurring between a first time and a second time; determining a backward increment between data on the data storage system the second time and data on the data storage system at the first time based on the plurality of data writes from the application program to the data storage system; storing the backward increment; storing the plurality of data writes; and updating the backup storage system so that the data on the data storage system at the second time is the same as the data on the backup storage system at the second time."

Goldstein never "receives a plurality of data writes from an application program" as asserted by the Examiner. Rather, unlike taking snapshots of a consistent state of data in the data volume disk storage, while system operation is briefly suspended, as taught by *Goldstein* (see col. 3, lines 43-55), the claims set forth in the present application teach intercepting data writes on their way to storage, without suspending operation (See FIG. 3 and specification [0056] through [0089]).

The two snapshots taught by *Goldstein* are virtual copies of the disk volume at two different times, not a plurality of data writes that are captured from an application program occurring between a first time and a second time, as set forth, in part, in claim 1. Neither the snapshots nor the difference between the snapshots are sufficient to enable the recovery of data to any point in time based on "receiving a plurality of data writes from an application program," as set forth, in part, in claim 1. A snapshot of a disk is static capture of the entire contents of a volume at a single point in time, not a "plurality of data writes from an application program to the data storage system."

While various succedent backups may be made using the difference list and a full base state backup (see *Goldstein* col. 4, lines 26 through col. 5 lines 22), the succedent backups are not accomplished by "determining a backward increment between data on the data storage system at the second time and data on the data storage system at the first time based on the plurality of data writes from the application program to the data storage system" as set forth in claim 1. Rather, *Goldstein* teaches incremental backups based on snapshots and differences between the snapshots, which cannot be equated with the backward

increments between data based on the plurality of data writes, as set forth, in part, in claim 1 of the present application.

Further, *Goldstein* fails to teach "storing the backward increment; storing the plurality of data writes; and updating the backup storage system so that the data on the data storage system at the second time is the same as the data on the backup storage system at the second time," as set forth, in part, in claim 1 of the present application. The reverse increment can be archived for later use (see specification [0036]). Instead, *Goldstein* teaches prior art snapshot techniques (see col. 3, lines 44-55) to take snapshots as backups. The difference in data blocks among various snapshots is then utilized to generate succedent and precedent backups. Backward increments are not stored. As discussed herein, *Goldstein* fails to receive a plurality of data writes, which thus cannot be stored in *Goldstein*. In fact, *Goldstein* indicates that the snapshots are typically deleted, once they are rolled out (see FIGS. 5 and 6 and col. 5 line 50 to col. 6 line 5).

Accordingly, *Goldstein* neither teaches nor suggests the subject matter set forth in claim 1, but rather teaches utilizing snapshots of consistent data states and the differences between those data states to provide succedent and precedent backups.

For at least these reasons, Applicant respectfully submits that *Goldstein* does not anticipate claim 1. The anticipation criterion has simply not been met by *Goldstein*.

Claim 20

Claim 20 recites "receiving a plurality of data writes captured between an application and the data storage system, the plurality of data writes occurring

between a first time and a second time; identifying data blocks in the data storage system that were changed based on the plurality of data writes; applying the plurality of data writes to an image on the backup storage system; determining a forward increment between data on the data storage system at the first time and data on the data storage system at the second time based on the plurality of data writes; and determining a backward increment between data on the data storage system at the second time and data on the data storage system at the first time based on a plurality of data writes; ." As discussed herein, with respect to claim 1, *Goldstein* fails to teach "receiving a plurality of data writes captured between an application and the data storage system, the plurality of data writes occurring between a first time and a second time."

The two snapshots taught by *Goldstein* are virtual copies of the disk volume at those two different times, not a plurality of data writes captured between an application and the data storage system, and occurring between a first time and a second time. *Goldstein* compares the two snapshots to determine a difference list, based on the difference between the two snapshots, as discussed herein.

While *Goldstein* does describe identifying data blocks in the data storage system that changed based on the difference between the two snapshots, *Goldstein* fails to teach "identifying data blocks in the data storage system that were changed based on the plurality of data writes." As in the example of the two pictures of the chair, the data blocks that change between two snapshots is entirely different from the data blocks that change based on the plurality of data writes, or based on a video of the chair actually being moved. As another example, if a data block is written more than once between the time of the first

snapshot and the time of the second snapshot, *Goldstein* indicates the last data that was written to the data block, rather than "identifying data blocks in the data storage system that were changed based on the plurality of data writes," as set forth in claim 20.

Goldstein also fails to teach "applying the plurality of data writes to an image on the backup storage system." Specifically, *Goldstein* typically deletes or overwrites the snapshots taken based on new snapshots (see FIGS. 5 and 6). Thus, in addition to the fact that *Goldstein* fails to teach "the plurality of data writes," *Goldstein* appears to apply only current snapshots to any base snapshot it has stored. "To recover the data volume in the present example, the backups are restored in successive order. The full base state backup 130 (B.sub.0) is obtained and subsequently overwritten with the first succedent backup 131 and then with the second succedent backup 133. This yields an exact copy of the volume as of its second state snapshot 115" (see col. 5, lines 50-54). Further, *Goldstein* teaches concatenating various snapshots. Combining snapshots of a disk volume, even with the difference list, is not the same as "applying the plurality of data writes to an image on the backup storage system," as set forth in claim 20.

Finally, *Goldstein* fails to teach "determining a forward increment between data on the data storage system at the first time and data on the data storage system at the second time based on the plurality of data writes; determining a backward increment between data on the data storage system at the second time and data on the data storage system at the first time based on a plurality of data writes; storing the forward increment; storing the backward increment; storing the plurality of data writes; and updating the backup storage system so that the data on the data storage system at the second time is the same

as the data on the backup storage system at the second time." Assuming that "the forward increment" may be determined from the difference list the, *Goldstein* determines a "forward increment" based on the difference between the two snapshots, not based on the plurality of data writes. Assuming that "the backward increment" may be determined from the difference list the, *Goldstein* determines a "backward increment" based on the difference between the two snapshots, not based on the plurality of data writes, as discussed herein. Thus, any increments determined in *Goldstein* do not indicate what happened to the data between the snapshots and thus cannot be utilized to recover the data to any point in time. Further, any increments discussed in *Goldstein* are not stored, but are overwritten, as discussed herein.

In the present application, a reverse increment can include individual data writes that can be applied to the application's image--i.e., the production image, replicated image, or read/write snapshot of the replicated or production images, as one transaction to restore the image from the second time back to the first time. The forward increment can then be applied in part, or in its entirety as one transaction to roll the image forward from the first time to any point in time up to and including the second time. Specification [0036]. Thus, while the production image in the present application may include a snapshot, the reverse or forward increments comprising individual data writes, unlike the additional snapshots and difference lists taught by *Goldstein*, are utilized to roll the production image forward or backward to any point in time, rather than to a time determined according to when the snapshot in *Goldstein* is taken.

For at least these reasons, Applicant respectfully submits that *Goldstein* does not anticipate claim 20. The foregoing anticipation criterion has simply not been met by *Goldstein*.

In view of the remarks set forth herinabove, all of the independent claims are deemed allowable, along with any claims depending therefrom.

Respectfully submitted,

Gregory A. Becker et al.

Date: 2/15/07 By: Kenneth M. Kaslow

Kenneth M. Kaslow, Reg. No. 32,246
Carr & Ferrell *LLP*
2200 Geng Road
Palo Alto, CA 94303
TEL: (650) 812-3465
FAX: (650) 812-3444

VIII. CLAIMS APPENDIX (37 C.F.R. § 41.37(c)(1)(viii))

Claims on Appeal:

1. A method for maintaining a backup storage system for a data storage system comprising:

receiving a plurality of data writes from an application program, the plurality of data writes occurring between a first time and a second time;

determining a backward increment between data on the data storage system at the second time and data on the data storage system at the first time based on the plurality of data writes from the application program to the data storage system;

storing the backward increment;

storing the plurality of data writes; and

updating the backup storage system so that the data on the data storage system at the second time is the same as the data on the backup storage system at the second time.

2. The method of claim 1, further comprising:

determining a forward increment between the data on the data storage system at the first time and the data on the data storage system at the second time based on the plurality of data writes.

3. The method of claim 2, further comprising:

associating the backward increment with the forward increment.

4. The method of claim 2, further comprising:

storing the forward increment; and

storing an association of the backward increment and the forward increment.

5. The method of claim 1, further comprising:

storing indicia of the plurality of data writes.

6. The method of claim 1, wherein said updating the backup storage system comprises:

applying each of the plurality of data writes to an image of data on the backup storage system, thereby recreating the data on the data storage system at the second time.

7. The method of claim 6, said applying each of the plurality of data writes comprising:

updating the image of the data stored on the backup storage system with the plurality of data writes.

8. The method of claim 1, wherein said updating the backup storage system comprises:

optimally applying the plurality of data writes to the backup storage system, thereby recreating the data on the data storage system at the second time.

9. The method of claim 1, wherein a difference between the first time and the second time is a predetermined time period.

10. The method of claim 1, wherein a difference between the first time and the second time is a variable time period.

11. The method of claim 10, wherein a difference between the first time and the second time is dependent on the rate of the plurality of data writes.

12. The method of claim 7, wherein a difference between the first time and the second time is dependent on a quantity of the plurality of data writes.

13. (cancelled)

14. The method of claim 1, wherein said updating the backup storage system so that the data on the data storage system at the second time is the same as the data on the backup storage system at the second time includes applying the backward increment to an image of data on the backup storage system, thereby recreating the data on the data storage system at the second time.

15. The method of claim 14, wherein said updating the backup storage system so that the data on the data storage system at the second time is the same as the data on the backup storage system at the second time includes applying an individual data write to the image of data on the backup storage system, thereby recreating the data on the data storage system at a point in time between the first time and the second time.

16-19. (cancelled)

20. A method for using a backup storage system for a data storage system comprising:

receiving a plurality of data writes captured between an application and the data storage system, the plurality of data writes occurring between a first time and a second time;

identifying data blocks in the data storage system that were changed based on the plurality of data writes;

applying the plurality of data writes to an image on the backup storage system;

determining a forward increment between data on the data storage system at the first time and data on the data storage system at the second time based on the plurality of data writes;

determining a backward increment between data on the data storage system at the second time and data on the data storage system at the first time based on a plurality of data writes;

storing the forward increment;

storing the backward increment;

storing the plurality of data writes; and

updating the backup storage system so that the data on the data storage system at the second time is the same as the data on the backup storage system at the second time.

21-25. (cancelled)

26. A system for maintaining a backup storage system for a data storage system comprising:

a production intercept layer configured to receive a plurality of data writes from an application program, the plurality of data writes occurring between a first time and a second time;

a backup agent configured to determine a backward increment between data on the data storage system at the second time and data on the data storage system at the first time based on the plurality of data writes from the application program to the data storage system;

a log file container configured to store the backward increment;

a storage device configured to store the plurality of data writes; and

a backup manager configured to update the backup storage system so that the data on the data storage system at the second time is the same as the data on the backup storage system at the second time.

27. The system of claim 26, wherein the backup agent is further configured to determine a forward increment between the data on the data storage system at the first time and the data on the data storage system at the second time based on the plurality of data writes.

28. The system of claim 26, wherein a difference between the first time and the second time is a predetermined time period.

29. The system of claim 26, wherein a difference between the first time and the second time is a variable time period.

30. The system of claim 29, wherein a difference between the first time and the second time is dependent on the rate of the plurality of data writes.

31. The system of claim 26, wherein the backup manager is further configured to apply the backward increment to an image of data on the backup storage system, thereby recreating the data on the data storage system at the second time.

32. The system of claim 26, wherein the backup manager is further configured to apply an individual data write to the image of data on the backup storage system, thereby recreating the data on the data storage system at a point in time between the first time and the second time.

IX. EVIDENCE APPENDIX (37 C.F.R. § 41.37(c)(1)(ix))

There is no such evidence.

X. RELATED PROCEEDING APPENDIX (37 C.F.R. § 41.37(c)(1)(x))

N/A